**SVEUČILIŠTE U ZAGREBU**

**PRIRODOSLOVNO–MATEMATIČKI FAKULTET**

**MATEMATIČKI ODSJEK**

Ivan Čeh

# UBRZANJE QR FAKTORIZACIJE S PIVOTIRANJEM

Diplomski rad

Voditelj rada:
prof. dr. sc. Sanja Singer

Zagreb, studeni, 2018.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

# Sadržaj

# Uvod

Jedna od najvažnijih faktorizacija u području numeričke linearne algebre je QR faktorizacija. Njezine najpoznatije primjene su rješavanje sustava linearnih jednadžbi, metoda najmanjih kvadrata, a nezamjenjiva je i kao dio algoritama za određivanje svojstvenih vrijednosti matrice. Za neke se primjene, poput određivanja ranga matrice i aproksimaciju matrice matricom nižeg ranga koristi QR faktorizacija sa stupčanim pivotiranjem (QRCP). Da bi se stupčano pivotiranje provelo, potrebno je pamtiti norme stupaca i po potrebi ih ažurirati. Izbor sljedećeg pivotnog stupca (tj. stupca s najvećom normom preostalog dijela stupca) znatno usporava algoritam. Pritom najveći problem predstavlja komunikacija sa sporom memorijom. Zato je u interesu znanstvenika razviti metodu koja bi zamijenila QRCP i dala podjednako kvalitetne rezultate te na računalu imala slične performanse kao standardna QR faktorizacija.

Bischof predlaže metodu kontroliranog pivotiranja unutar lokalnih blokova [2], Demmel i suradnici predlažu metodu s izbjegavanjem komunikacije [5], a često se koristi i randomizirana QR faktorizacija [7, 9]. U ovom radu opisat ćemo navedene algoritme te ih usporediti na 256-dretvenom računalu Xeon Phi.

# Poglavlje 1

# QR faktorizacija, osnovni algoritmi

## 1.1 Standardna QR faktorizacija

**Definicija 1.1.1.** *Za realnu matricu $A \in \mathbb{R}^{m\times n}$ $(m \geq n)$ kažemo da ima ortogonalne stupce ako vrijedi $A^TA = I$.*

Jednostvanije rečeno, matrica ima ortogonalne stupce ako joj stupci čine ortonormiran skup, tj. skalarni produkt svaka dva različita stupaca je 0, a norma svakog stupca je 1. Ako je $m = n$, onda je $A^T = A^{-1}$, pa vrijedi i

$$AA^T = AA^{-1} = I$$

Kvadratne matrice s ortogonalnim stupcima zovemo ortogonalnim matricama. Sada navedimo neke korisne činjenice o matricama s ortogonalnim stupcima.

**Propozicija 1.1.2.** *Neka su $A \in \mathbb{R}^{m\times n}$ i $B \in \mathbb{R}^{n\times l}$, pri čemu je $m \geq n \geq l$, matrice s ortogonalnim stupcima. Tada je i njihov produkt $AB \in \mathbb{R}^{m\times l}$ matrica s ortogonalnim stupcima.*

*Dokaz.* Iz definicije izlazi

$$(AB)^T(AB) = B^TA^TAB = B^TI_nB = B^TB = I_l. \qquad \square$$

**Propozicija 1.1.3.** *Neka je $A \in \mathbb{R}^{m\times n}$, $m \geq n$ matrica s ortogonalnim stupcima i $x \in \mathbb{R}^n$ proizvoljan vektor. Tada vrijedi*

$$||Ax||_2 = ||x||_2,$$

*pri čemu $||\ ||_2$ označava 2-normu,*

$$||x||_2 = \sqrt{x^Tx}.$$

*Dokaz.* Iz definicije 2-norme izlazi

$$||Ax||_2^2 = (Ax)^T(Ax) = x^TA^TAx = x^TIx = x^Tx = ||x||_2^2. \qquad \square$$

**Definicija 1.1.4.** *Za realnu matricu $A \in \mathbb{R}^{m\times n}$ kažemo da je gornjetrokutasta (u slučaju $m \neq n$ koristi se i naziv gornjetrapezoidna) ako za sve elemente $A_{ij}$ za koje je $i > j$ vrijedi $A_{ij} = 0$.*

*Za realnu matricu $A \in \mathbb{R}^{m\times n}$ kažemo da je donjetrokutasta (u slučaju $m \neq n$ koristi se i naziv donjetrapezoidna) ako za sve elemente $A_{ij}$ za koje je $i < j$ vrijedi $A_{ij} = 0$.*

Sada navodimo bez dokaza još neke tvrdnje koje vrijede za gornjetrokutaste i donjetrokutaste matrice.

**Propozicija 1.1.5.** *Produkt dviju gornjetrokutastih (donjetrokutastih) matrica je gornjetrokutasta (donjetroutasta) matrica.*

**Propozicija 1.1.6.** *Gronjetrokutasta (donjetrokutasta) kvadratna matrica je nesingularna ako i samo ako su joj svi elementi na dijagonali različiti od* 0*. Tada je njezin inverz gornjetrokutasta (donjetrokutasta) matrica.*

**Definicija 1.1.7.** *Neka je $A \in \mathbb{R}^{m\times n}$ matrica gdje je $m \geq n$. QR faktorizacija ili QR dekompozicija matrice A je rastav*

$$A = QR,$$

*gdje je $Q \in \mathbb{R}^{m\times l}$ matrica s ortogonalnim stupcima, a $R \in \mathbb{R}^{l\times n}$ gornjetrokutasta matrica.*

*Ako je $l = n$ tada govorimo o skraćenoj QR faktorizaciji, a ako je $l = m$ tada govorimo o punoj QR faktorizaciji.*

## Gram–Schmidtov postupak ortogonalizacije

**Teorem 1.1.8.** *Neka je $A \in \mathbb{R}^{m\times n}$ gdje je $m \geq n$ i* $\operatorname{rang}(A) = n$*. Tada postoji jedinstvena faktorizacija oblika*

$$A = QR,$$

*gdje je $Q \in \mathbb{R}^{m\times n}$ matrica s ortogonalnim stupcima, a $R \in \mathbb{R}^{n\times n}$ gornjetrokutasta s pozitivnim elementima na dijagonali.*

*Dokaz.* Tvrdnju dokazujemo Gram–Schmidtovim postupkom ortogonalizacije (vidjeti algoritam 1).

Algoritam dokazuje postojanje tražene faktorizacije, ali pažljivim promatranjem vidimo da su svi elementi matrica $Q$ i $R$, zapravo, jedini mogući izbor, čime dobivamo jedinstvenost. □

**Algoritam 1** QR faktorizacija Gram–Schmidtovim postupkom ortogonalizacije

**Require:** $A \in \mathbb{R}^{m\times n}$

$A = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}$ $\{a_1, a_2, \ldots, a_n$ su vektori stupci matrice $A\}$

**for** $i = 1$ **to** $n$ **do**

$b_i \leftarrow a_i \sum_{j=1}^{n-1} \langle a_i, q_j \rangle q_j$

$q_i \leftarrow \dfrac{b_i}{\|b_i\|_2}$ {Uvjet rang$(A) = n$ osigurava $\|b_i\|_2 \neq 0$}

**for** $j = 1$ **to** $i - 1$ **do**

$r_{ij} \leftarrow \langle a_i, q_j \rangle$

**end for**

$r_{ii} \leftarrow \|b_i\|_2$

**end for**

$Q = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix}$ $\{q_1, q_2, \ldots, q_n$ su vektori stupci matrice $Q\}$

$$R = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{nn} \end{bmatrix}$$

## Householderovi reflektori

Gram–Schmidtov postupak koristan je za razvoj teorije, no u praksi se rijetko koristi za QR faktorizaciju zbog numeričke nestabilnosti. Naime, često je dobivena matrica $Q$ daleko od ortogonalne. Najčešća dva algoritma koja se koriste u praksi su Givensove rotacije i Householderovi reflektori. Koncentrirat ćemo se na QR faktorizaciju Householderovim reflektorima.

**Definicija 1.1.9.** *Neka je $v \in \mathbb{R}^n$ vektor 2-norme 1. Householderov reflektor je linearni operator definiran sa $H_v := I - 2vv^T$.*

Naziv "reflektor" dolazi iz prostorne predodžbe. Naime, operator $H$, zapravo, reflektira vektor $v$ s obzirom na hiperravninu okomitu na vektor $v$.

**Propozicija 1.1.10.** *Householderovi reflektori su simetrične i ortogonalne matrice.*

*Dokaz.* Simetričnost vrijedi jer je:

$$H_v^T = (I - 2vv^T)^T = I^T - 2(vv^T)^T = I - 2vv^T = H_v.$$

Ortogonalnost vrijedi jer je:

$$\begin{aligned} H_v H_v^T &= H_v H_v = (I - 2vv^T)(I - 2vv^T) = I - 2vv^T - 2vv^T + 4vv^T vv^T \\ &= I - 4vv^T + 4v\|v\|_2^2 v^T = I - 4vv^T + 4vv^T = I. \end{aligned}$$ □

Za QR faktorizaciju treba konstruirati Householderov reflektor koji poništava sve elemente vektora osim prvog, tj. takav da za vektor $x \in \mathbb{R}^n$ vrijedi

$$Hx = \alpha e_1.$$

Zbog Propozicije 1.1.3 odmah slijedi da je

$$\|x\|_2 = \|Hx\|_2 = \|\alpha e_1\|_2 = |\alpha| \, \|e_1\|_2 = |\alpha|.$$

Pokazuje se da takav reflektor $H$ dobivamo odabirom

$$v = \frac{x \pm \|x\|_2 e_1}{\|x \pm \|x\|_2 e_1\|_2}.$$

Zbog numeričke stabilnosti, uobičajeno se uzima predznak kojeg ima prva komponenta vektora $x$, tako da ne dođe do katastrofalnog kraćenja. Taj izbor se zapisuje kao

$$v = \frac{x + \operatorname{sign}(x_1)\|x\|_2 e_1}{\|x + \operatorname{sign}(x_1)\|x\|_2 e_1\|_2}.$$

Bez obzira na izbor predznaka, vrijedi

$$\begin{aligned} H_v x &= x - 2\frac{(x \pm \|x\|_2 e_1)(x \pm \|x\|_2 e_1)^T}{(x \pm \|x\|_2 e_1)^T (x \pm \|x\|_x e_1)} x = x - 2\frac{(x \pm \|x\|_2 e_1)(x^T x \pm \|x\|_2 x_1)}{x^T x \pm \|x\|_2 x_1 \pm \|x\|_2 x_1 + x^T x} \\ &= x - 2\frac{(x \pm \|x\|_2 e_1)(x^T x \pm \|x\|_2 x_1)}{2(x^T x \pm \|x\|_2 x_1)} = x - (x \pm \|x\|_2 e_1) = \mp \|x\|_2 e_1. \end{aligned}$$

Samo provođenje QR faktorizacije reflektorima je vrlo jednostavno. Prvo se prvi stupac svede samo na prvi element, a nakon toga postupak se provodi na podmatrici (bez prvog retka i stupca) transformirane matrice. Postupak se ponavlja dok se ne iscrpe svi stupci.

Primijetimo da se jednom dobivene nule u vodećim stupcima neće pokvariti primjenom sljedećih reflektora.

**Algoritam 2** QR faktorizacija korištenjem Householderovih reflektora

**Require:** $A \in \mathbb{R}^{m \times n}$
  **for** $i = 1$ **to** $n$ **do**
    $x \leftarrow A_{i:n,i}$
    $v \leftarrow \dfrac{x \pm \|x\|_2 e_1}{\|x \pm \|x\|_2 e_1\|_2}$
    $H_i \leftarrow \begin{bmatrix} I & 0 \\ 0 & I - 2vv^T \end{bmatrix}$
    $A \leftarrow H_i A$
  **end for**
  $R = A$
  $Q \leftarrow H_1 H_2 \dots H_n$ {Q je ortogonalna po propoziciji 1.1.2}

Bitno je primijetiti da se djelovanje Householderovim reflektorom ne računa eksplicitnim kreiranjem matrice operatora $H$, već se na jedan vektor primjenjuje ovako:

$$Hx = x - v(v^T x),$$

odnosno na matricu $A$ se primjenjuje ovako:

$$HA = A - v(v^T A).$$

Dakle, djelovanje Householderova reflektora na vektor je operacija linearne vremenske složenosti (svodi se na operacije između 2 vektora, poput skalarnog produkta), dok je djelovanje Householderovog reflektora na matricu operacija kvadratne vremenske složenosti (svodi se na operacije između vektora i matrice, poput množenja vektora matricom). Radi se o "level 1 BLAS", odnosno "level 2 BLAS" operacijama biblioteke BLAS. U pravilu se veća efikasnost postiže povećanom upotrebom "level 3 BLAS" operacija poput matričnog množenja jer su one optimizirane tako da bolje iskorištavaju priručnu (cache) memoriju i paralelizam računala. Zato je bolje koristiti algoritam koji veću količinu posla prebacuje na "level 3 BLAS" funkcije.

## WY reprezentacija i blokovski algoritmi

Bischof i Van Loan su u [3] dali blokovsku QR faktorizaciju. Kompozicija Householderovih reflektora nije Householderov reflektor, ali pomaže činjenica da kompoziciju $k$ Householderovih reflektora možemo zapisati kao $I - WY^T$ gdje su $W, Y \in \mathbb{R}^{m \times k}$ i $Y$ je donjetrokutasta. Navedeni zapis naziva se WY reprezentacija produkta Householderovih reflektora. Kasnije su Schreiber i Van Loan to dodatno optimizirali pokazavši da se isti

produkt može zapisati i kao $I - YTY^T$ gdje je $Y \in \mathbb{R}^{m\times k}$ donjetrokutasta i $T \in \mathbb{R}^{k\times k}$ gornjetrokutasta. To omogućava spremanje matrica $Y$ i $T$ u jedno dvodimenzionalno polje dimenzija $(m+1)\times k$.

Neka je matrica $Y \in \mathbb{R}^{m\times k}$ donjetrokutasta, a $T \in \mathbb{R}^{k\times k}$ gornjetrokutasta. Takve ćemo operatore odsad nazivati blok-reflektorima. Time dobivamo algoritam u kojem je veća količina posla prebačena na "level 3 BLAS" rutinu DGEMM (double generic matrix multiply) što ubrzava algoritam.

**Algoritam 3** Blokovska QR faktorizacija korištenjem Householderovih reflektora

**Require:** $A \in \mathbb{R}^{m\times n}$
**Require:** $b \in \mathbb{N}$ je veličina bloka
  **for** $i = 0, b, 2b$ **to** broj blokova $\cdot\, b$ **do**
    **for** $j = i+1$ **to** $\min\{i+b, n\}$ **do**
      Primijeni trenutni blok-reflektor iz bloka na $j$-ti stupac
      Poništi sve poddijagonalne elemente u $j$-tom stupcu reflektorom $H_j$
      Nadopuni blok-reflektor reflektorom $H_j$
    **end for**
    Primijeni blok-reflektor iz trenutnog bloka na podmatricu $A_{i+1:m,i+b+1:n}$
  **end for**
  $R = A$
  $Q$ je kompozicija svih blok-reflektora {Q je ortogonalna po propoziciji 1.1.2}

Za provedbu blokovskog QR algoritma korištenjem kompaktne WY reprezentacije potrebno je znati generirati matrice $Y$ i $T$ iz danih vektora $v_i$ koji generiraju reflektore.

**Algoritam 4** Generiranje $Y$ i $T$ matrica za kompaktnu WY reprezentaciju kompozicije Householderovih reflektora

**Require:** $v_1, v_2, \ldots v_k$ vektori $k$ Householderovih reflektora na $\mathbb{R}^m$
  $Y \leftarrow \begin{bmatrix} v_1 \end{bmatrix}$
  $T \leftarrow \begin{bmatrix} -2 \end{bmatrix}$
  **for** $i = 2$ **to** $k$ **do**
    $z \leftarrow -2TY^T v_j$
    $Y \leftarrow \begin{bmatrix} Y & v_i \end{bmatrix}$
    $T \leftarrow \begin{bmatrix} T & z \\ 0 & -2 \end{bmatrix}$
  **end for**

## 1.2 QR faktorizacija sa stupčanim pivotiranjem

Kao što je u uvodu već navedeno, katkad je, zbog stabilnosti, QR faktorizaciju potrebno računati korištenjem stupčanog pivotiranje. Prvo definiramo što je permutacijska matrica.

**Definicija 1.2.1.** *Permutacijska matrica je kvadratna binarna matrica koja u svakom retku i svakom stupcu točno na jednom mjestu ima element* 1*, a na svim ostalim mjestima ima elemente* 0*.*

Permutacijska matrica se tako naziva zbog svog djelovanja na vektore i matrice. Naime, neka je

$$\pi = \begin{pmatrix} 1 & 2 & \cdots & n \\ \pi(1) & \pi(2) & \cdots & \pi(n) \end{pmatrix}$$

neka permutacija. Pripadna permutacijska matrica je definirana ovako:

$$P_{ij} = \begin{cases} 1, & \text{ako je } \pi(i) = j, \\ 0, & \text{inače.} \end{cases}$$

Za nju vrijedi:

1. Ako je $x \in \mathbb{R}^n$ vektor, tada je $(Px)_i = x_{\pi(i)}$, za sve $i$ od 1 do $n$.
2. Ako je $A \in \mathbb{R}^{n\times m}$ matrica, vrijedi $(PA)_{i,1:n} = A_{\pi(i),1:n}$. Dakle, množenje s permutacijskom matricom slijeva mijenja poredak redaka.
3. Ako je $A \in \mathbb{R}^{m\times n}$ matrica, vrijedi $(AP)_{1:n,\pi(i)} = A_{1:n,i}$. Dakle, množenje s permutacijskom matricom zdesna mijenja poredak stupaca.

**Definicija 1.2.2.** *Neka je $m \geq n$ i $A \in \mathbb{R}^{m\times n}$ matrica. QR faktorizacija matrice $A$ sa stupčanim pivotiranjem (QRCP) je rastav*

$$A = QRP^T,$$

*odnosno*

$$AP = QR,$$

*gdje je $P \in \mathbb{R}^{n\times n}$ permutacijska matrica, $Q \in \mathbb{R}^{m\times l}$ matrica s ortogonalnnim stupcima i $R \in \mathbb{R}^{l\times n}$ gornjetrokutasta. Ako je $l = n$ radi se o skraćenoj QRCP faktorizaciji, a ako je $l = m$ radi se o punoj QRCP faktorizaciji.*

Obično QRCP faktorizaciju radimo tako da od preostalih stupaca izabiremo onaj koji ima najveću (najčešće 2-) normu radnog dijela stupca. Takav stupac dovodi se na mjesto koje se prvo transformira. Time elementi na dijagonali matrice $R$ nužno padaju po apsolutnoj vrijednosti. Čak štoviše, vrijedi i

$$|R_{ii}| \geq \|R_{i:n,j}\|_2$$

za sve indekse $j \geq i$. To, osim veće numeričke stabilnosti, QRCP faktorizaciji daje i nove primjene.

**Algoritam 5** QRCP faktorizacija korištenjem Householderovih reflektora

**Require:** $A \in \mathbb{R}^{m\times n}$
  $P \leftarrow I_n$
  $N_i \leftarrow \|A_{1:n,i}\|_2$ {Odredimo početne norme stupaca}
  **for** $i = 1$ **to** $n$ **do**
    Tražimo indeks $j$ između $i$ i $n$ za koji je $N_i$ najveća. $\{N_j = \|A_{i:n,j}\|_2\}$
    U matricama $A$ i $P$ zamijenimo $i$-ti i $j$-ti stupac.
    $x \leftarrow A_{i:n,i}$
    $v \leftarrow \dfrac{x \pm \|x\|_2 e_1}{\|x \pm \|x\|_2 e_1\|_2}$
    $H_i \leftarrow \begin{bmatrix} I & 0 \\ 0 & I - 2vv^T \end{bmatrix}$
    $A \leftarrow H_i A$
    Za sve $j$ od $i$ do $n$ ažuriramo $N_i$ tako da maknemo utjecaj elementa $A_{ij}$ na tu normu.
  **end for**
  $R = A$
  $Q \leftarrow H_1 H_2 \dots H_n$ {$Q$ ima ortogonalne stupce po propoziciji 1.1.2}

Standardna QR faktorizacija ponekad daje loše rezultate ili nije primjenjiva za matrice koje nisu punog stupčanog ranga ili su blizu neke matrice nižeg ranga. QRCP često pomaže u takvim situacijama. QRCP faktorizacija se, uz singularnu dekompoziciju, ili kraće SVD, često koristi i u računanju aproksimacija matricama nižeg ranga.

## 1.3 Primjene u aproksimacijama matricama nižeg ranga

Da bismo iskazali teoreme o aproksimacijama matricama nižeg ranga, potrebna nam je precizna definicija singularne dekompozicije.

**Definicija 1.3.1.** *Neka je $A \in \mathbb{R}^{m\times n}$, $m \geq n$ zadana matrica. Singularna dekompozicija (SVD) matrice A je rastav*

$$A = U\Sigma V^T,$$

*gdje je $\Sigma \in \mathbb{R}^{m\times n}$ dijagonalna matrica s nenegativnim dijagonalnim elementima, a $U \in \mathbb{R}^{m\times m}$ i $V \in \mathbb{R}^{n\times n}$ su ortogonalne matrice.*

*Elementi na dijagonali matrice $\Sigma$ nazivaju se singularne vrijednosti, a broj pojavljivanja pojedine singularne vrijednosti na dijagonali naziva se kratnost singularne vrijednosti.*

*Skraćena singularna dekompozicija definira se kao produkt*

$$A = U\Sigma V^T,$$

*gdje je* $\Sigma \in \mathbb{R}^{n\times n}$ *dijagonalna matrica s nenegativnim dijagonalnim elementima,* $U \in \mathbb{R}^{m\times n}$ *je matrica s ortogonalnim stupcima, a* $V \in \mathbb{R}^{n\times n}$ *je ortogonalna matrica.*

*Ako je* $m < n$*, onda se SVD matrice A dobiva iz SVD-a matrice* $A^T$*.*

Pokazano je da svaka matrica ima SVD, što omogućuje njezinu primjenu u općenitom slučaju.

**Definicija 1.3.2** (Matrične norme)**.** *Frobeniusova norma matrice* $A \in \mathbb{R}^{m\times n}$ *definira se ovako*

$$\|A\|_F := \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2}.$$

*Spektralna norma matrice* $A \in \mathbb{R}^{m\times n}$ *definira se ovako*

$$\|A\|_2 := \max_{x\in\mathbb{R}^n,\ \|x\|_2=1} \|Ax\|_2.$$

Pokazuje se da je spektralna norma matrice jednaka njezinoj najvećoj singularnoj vrijednosti.

Jedna od najčešćih primjena SVD-a i QRCP faktorizacije je određivanje aproksimacije matricama nižeg ranga. U teoremu 1.3.3 vidjet ćemo da se iz SVD-a lako računa najbolja aproksimacija nižeg ranga. QRCP faktorizacija ne garantira najbližu aproksimaciju, ali najčešće daje relativno dobru aproksimaciju uz znatno lakše računanje.

**Teorem 1.3.3** (Eckart-Young-Mirsky)**.** *Neka je* $A \in \mathbb{R}^{m\times n}$ *matrica i* $U\Sigma V^T$ *njezin SVD takv da su elementi na dijagonali matrice* $\Sigma$ *poredani u padajućem redoslijedu. Neka je* $r \in \mathbb{N}$*,* $r \leq \min\{m, n\}$*. Tada je*

$$A' = U_{1:m,1:r}\Sigma_{1:r,1:r}(V_{1:n,1:r})^T$$

*najbliža aproksimacija matrice A matricom ranga r, tj. za svaku drugu matricu* $A'' \in \mathbb{R}^{m\times n}$ *ranga r vrijedi* $\|A'' - A\|_F \geq \|A' - A|_F$ *i* $\|A'' - A\|_2 \geq \|A' - A\|_2$*.*

Dakle, najbližu aproksimaciju ranga $r$ dobivamo tako da u obzir uzmemo samo najvećih $r$ singularnih vrijednosti, a ostale odbacimo, tj. zamijenimo ih nulama.

Slično, ako je $P^T QR$ neka QRCP faktorizacija matrice $A \in \mathbb{R}^{m\times n}$ tada za aproksimaciju ranga $r$ koristimo matricu

$$A' = P^T Q_{1:m,1:r} R_{1:r,1:n}.$$

S QRCP faktorizacijom nemamo teorijsku potvrdu da ćemo dobiti najbolju aproksimaciju no eksperimentalno je pokazano da je aproksimacija uglavnom poprilično dobra.

Dobro je to što u takvim primjenama faktorizaciju ne moramo provoditi do kraja. Na primjer, u slučaju određivanja aproksimacije ranga $r$ dovoljno je odrediti prvih $r$ pivotnih stupaca i nakon njih zaustaviti algoritam. Tako dobivamo rastav

$$AP = \begin{bmatrix} Q_{11} & Q_{12} \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix},$$

gdje je $R_{11} \in \mathbb{R}^{r \times r}$ gornjetrokutasta. Pivotiranje je provedeno tako da norma podmatrice $R_{22}$ bude čim manja. Njezinim odbacivanjem dobivamo aproksimaciju

$$A' = P^T Q_{11} \begin{bmatrix} R_{11} & R_{12} \end{bmatrix}.$$

## 1.4 Blokovska QRCP faktorizacija

Algoritam blokovske QRCP faktorizacije prvi su predložili Quintana–Orti, Sun i Bischof u [10]. Sličan je algoritmu 3, a problem ažuriranja stupčanih normi rješava tako da u $j$-tom koraku primjenjuje trenutni blok-reflektor samo na $j$-ti redak kako bi imao vrijednosti koje su mu potrebne za ažuriranje normi stupaca.

**Algoritam 6** Blokovska QRCP faktorizacija korištenjem Householderovih reflektora

**Require:** $A \in \mathbb{R}^{m \times n}$
**Require:** $b \in \mathbb{N}$ je veličina bloka
  $N_i \leftarrow \|A_{1:n,i}\|_2$ {Odredimo početne norme stupaca}
  **for** $i = 0, b, 2b$ **to** broj blokova $\cdot\, b$ **do**
    **for** $j = i + 1$ **to** $\min\{i + b, n\}$ **do**
      Primijeni trenutni blok-reflektor iz bloka na $j$-ti stupac
      Poništi sve poddijagonalne elemente u $j$-tom stupcu reflektorom $H_j$
      Nadopuni blok-reflektor reflektorom $H_j$
    **end for**
    Primijeni blok-reflektor iz trenutnog bloka na podmatricu $A_{i+1:m,i+b+1:n}$
  **end for**
  $R = A$
  $Q$ je kompozicija svih blok-reflektora {$Q$ je ortogonalna po propoziciji 1.1.2}

Algoritam 6 donosi poboljšanje performansi u odnosu na algoritam 5 no i dalje je znatno sporiji od algoritma 3 zbog ažuriranja $j$-tog retka u svakom koraku.

# Poglavlje 2

# Napredni algoritmi

Algoritam 5 za QRCP faktorizaciju je za računanje neznatno zahtjevniji od algoritma 2. Primjena blokovskog algoritma 3 je značajno ubrzala računanje QR faktorizacije na modernim računalima, no analognu tehniku ne možemo primijeniti na računanje QRCP faktorizacije jer ne možemo unaprijed odrediti sljedećih $b$ pivotnih stupaca koji bi formirali blok (jer u svakom koraku treba ažurirati vrijednosti stupčanih normi). Zato je razvijeno nekoliko algoritama koji se na različite načine suočavaju s tim problemom kako bi povećali korištenje "BLAS level 3" rutina i time ubrzali QRCP faktorizaciju. U ovom poglavlju predstavljamo neke od njih.

## 2.1 Algoritam s kontroliranim lokalnim pivotiranjem

Christian H. Bischof je u [2] predložio blokovski algoritam s kontroliranim lokalnim pivotiranjem. Na pivotnom prozoru od $b$ stupaca provodimo QRCP faktorizaciju pomoću WY reprezentacije. Posebnost je u tome što u svakom koraku procjenjujemo uvjetovanost gornjetrokutaste matrice koju dobivamo dodavanjem novog stupca u matricu dosad iskorištenih stupaca. Kad se pokaže da je ta matrica loše uvjetovana tada sve preostale stupce trenutnog bloka "odbacujemo", tj. premještamo ih na najdesnija mjesta u matrici. Trenutni blok nadopunjujemo s novih $b$ stupaca na koje primijenimo trenutni blok-reflektor i nastavljamo postupak dok ne nađemo $b$ pivotnih stupaca. Za procjenu uvjetovanosti gornjetrokutaste matrice koristimo postupak inkrementalne procjene uvjetovanosti.

### Inkrementalna procjena uvjetovanosti

Algoritam inkrementalne procjene uvjetovanosti za trokutaste matrice je predstavio Bishof u [1]. Algoritam zapravo procjenjuje najmanju singularnu vrijednost matrice. Postoje brojni drugi algoritmi koji se bave tim problemom no posebnost navedenog algoritma je

u tome što omogućava praćenje najmanje singularne vrijednosti u situaciji kada matrici postupno dodajemo nove retke (u slučaju donjetrokutaste matrice) ili stupce (u slučaju gornjetrokutaste matrice). Algoritam je prilagođen donjetrokutastim matricama, a u slučaju gornjetrokutaste matrice $R$ algoritam primjenjujemo na donjetrokutastu matricu $R^T$ koja ima iste singularne vrijednosti.

Za najmanju singularnu vrijednost matrice $A$ vrijedi

$$\sigma_{\min(A)} = \frac{1}{\max\{\|x\|_2 : \|Ax\|_2 = 1\}}.$$

Inkrementalna procjena uvjetovanosti donjetrokutaste matrice $L$, zapravo, postupno kreira vektor $x$ tako da u svakom koraku vrijedi

$$\|L_{1:k,1:k} x_k\|_2 = 1$$

i $x_k$ ima čim veću normu. Za to koristimo pohlepnu tehniku opisanu u algoritmu 7.

**Algoritam 7** Inkrementalna procjena uvjetovanosti

**Require:** $L \in \mathbb{R}^{n\times n}$ je donjetrokutasta matrica

$x \leftarrow \dfrac{1}{a_{11}}$

**for** $i = 2$ **to** $n$ **do**

$v \leftarrow (L_{i,1:i-1})^T$ $\quad \{v \in \mathbb{R}^{i-1}$ je vektor$\}$

$\gamma \leftarrow L_{ii}$ $\quad \{\gamma \in \mathbb{R}\}$

**if** $\alpha = 0$ **then**

**if** $|\gamma|\,\|x\|_2 > 1$ **then**

$s \leftarrow 1; \quad c \leftarrow 0$

**else**

$s \leftarrow 0; \quad c \leftarrow 1$

**end if**

**else**

$\alpha \leftarrow v^T x; \quad \beta \leftarrow \gamma^2 x^T x + \alpha^2 - 1; \quad \eta \leftarrow \dfrac{\beta}{2\alpha}; \quad \mu \leftarrow \alpha\left(\eta + \operatorname{sign}(\alpha)\sqrt{\eta^2+1}\right)$

$\begin{bmatrix} s \\ c \end{bmatrix} \leftarrow \dfrac{1}{\sqrt{\mu^2+1}} \begin{bmatrix} \mu \\ -1 \end{bmatrix}$

**end if**

$x \leftarrow \begin{bmatrix} sx \\ \dfrac{c - s\alpha}{\gamma} \end{bmatrix}$

$kvadrat\_norme\_x \leftarrow kvadrat\_norme\_x + \dfrac{c - s\alpha}{\gamma}$

**end for**

Ideja algoritma je sljedeća. Neka je

$$L_{k+1} = \begin{bmatrix} L_k & 0 \\ v & \gamma \end{bmatrix}$$

i $x_k$ je neko rješenje jednadžbe

$$\|L_k x\|_2 = 1$$

takvo da je $\|x\|_2$ čim veća i neka je

$$d_k = L_k x_k.$$

Tada $x_{k+1}$ dobivamo kao rješenje jednadžbe

$$L_{k+1} x_{k+1} = \begin{bmatrix} s d_k \\ c \end{bmatrix},$$

gdje je $s^2 + c^2 = 1$. Dobivamo da mora biti

$$x_{k+1} = \begin{bmatrix} sx \\ \frac{c - s\alpha}{\gamma} \end{bmatrix},$$

gdje je $\alpha = v^T x$. Veličine $c$ i $s$ odabiremo tako da $\|x_{k+1}\|_2$ bude čim veća. Dobivamo da se najveća norma postiže za

$$\begin{bmatrix} s \\ c \end{bmatrix} = \frac{1}{\sqrt{\mu^2 + 1}} \begin{bmatrix} \mu \\ -1 \end{bmatrix},$$

gdje je $\mu$ definiran kao u algoritmu 7. Detaljnije objašnjenje može se naći u [1]. Bitno je uočiti da u $i$-tom koraku algoritma možemo lako doći do

$$\sigma_{\min}(L_k) \approx \frac{1}{\sqrt{kvadrat\_norme\_x}},$$

što znači da u svakom trenutku imamo procjenu uvjetovanosti dok matricu generiramo dodavajući po jedan redak ili stupac.

## Opis algoritma

Konačno, u algoritmu 8 opisan je postupak.

**Algoritam 8** QR faktorizacija s kontroliranim lokalnim pivotiranjem

**Require:** $A \in \mathbb{R}^{m\times n}$
**Require:** $b$ je veličina bloka
**Require:** *threshold* $\in \mathbb{R}$ je prag osjetljivosti, najmanja vrijednost $\sigma_{\min}$ koju dopuštamo
$P \leftarrow I_n$; $k \leftarrow 1$; $sr \leftarrow n+1$; $acc \leftarrow 0$
{$sr$ označava početak odbačenih stupaca (”start rejected”), a $acc$ je broj dosad prihvaćenih stupaca}
**while** $k < sr$ **do**
$kb \leftarrow \min\{n-k+1, b\}$
{$kb$ je veličina trenutnog bloka, jednaka $b$ osim ako je ostalo manje od $b$ stupaca}
$lacc \leftarrow 0$
{$lacc$ je broj dosad prihvaćenih stupaca iz trenutnog bloka (”locally accepted”)}
$nrj \leftarrow 0$
{$nrj$ je broj odbačenih stupaca u trenutnom bloku (”number of rejected”)}
$sl \leftarrow k+kb$; $ul \leftarrow k+kb$
{$sl$ je granica do koje tražimo kandidata za pivotni stupac (”search limit”), $ul$ je granica do koje ažuriramo stpupce trenutnim reflektorom (”update limit”)}
$res(i) = \|A_{k:m,i}\|_2$ za $i$ od $k$ do $k+kb-1$
**while** $lacc < kb$ **do**
$pvt \leftarrow i$, $k+lacc \le i \le sl$ za koji je $res(i)$ najveći
{odabir sljedećeg pivotnog stupca}
Primijeni algoritam 7 na gornjetrokutastu matricu
$\begin{bmatrix} A_{1:k+lacc-1,1:k+lacc-1} & A_{1:k+lacc-1,pvt} \\ 0 & res(pvt) \end{bmatrix}$.
{Izvršava se samo jedan korak petlje algoritma, jer je algoritam dosad već proveden na matrici $A_{1:k+lacc-1,1:k+lacc-1}$, te u memoriji pamtimo $x$ i njegovu normu}
Izvrši unutarnji dio algoritma prikazan u algoritmu 9
**end while**
Primijeni akumulirani blok-reflektor desno od pivotnog prozora (od $ul+1$-tog do $n$-tog stupca)
{Premjesti odbijene stupce na kraj matrice.}
$ti \leftarrow sr$
**for** $i = \max(ul-nrj, k+kb)$ **to** $\min ul-1, sr-nrj-1$ **do**
$ti \leftarrow ti-1$; $P_{i,1:n} \leftrightarrow P_{ti,1:n}$; $A_{1:n,i} \leftrightarrow A_{1:n,ti}$
**end for**
$srj \leftarrow srj-nrj$; $acc \leftarrow acc+\min(kb, ul-k-nrj)$; $k \leftarrow k+kb$
**end while**
Ako je potrebno, algoritmom 3 provedi QR faktorizaciju do kraja na matrici $A_{k:m,k:n}$

**Algoritam 9** Unutarnji dio algoritma QR faktorizacije s kontroliranim lokalnim pivotiranjem

**if** $\dfrac{1}{\|x\|} > threshold$

{Uzima se $x$ iz algoritma 7. Ako je uvjet zadovoljen stupac *pvt* je prihvaćen kao pivotni.}

**then**

$ti \leftarrow k + lacc$; $\quad lacc \leftarrow lacc + 1$

$P_{ti,1:n} \leftrightarrow P_{pvt,1:n}$; $\quad A_{1:m,ti} \leftrightarrow A_{1:m,pvt}$; $\quad res(ti) \leftrightarrow res(pvt)$

Generiraj novi Householderov reflektor, primijeni ga na preostale stupce pivotnog prozora (od *ti* do *ul*)

$res(j) \leftarrow \sqrt{res(j)^2 - A_{ti,j}^2}$ za $j$ od $ti + 1$ do $ul$

$$x \leftarrow \begin{bmatrix} sx \\ \hline \dfrac{c - s\alpha}{\gamma} \end{bmatrix}$$

**else**

{Svi preostali stupci pivotnog prozora su odbijeni. Povećavamo veličinu pivotnog prozora za još *kb* stupaca.}

$nl \leftarrow \min(sr - 1, ul + kb)$; $\quad nrj \leftarrow nrj + (sl - lacc)$

Primijeni blok-reflektor sakupljen na dosad prihvaćenim stupcima pivotnog prozora na stupce od $ul + 1$ do $nl$

{Pomakni odbijene stupce na najdesnija mjesta u povećanom pivotnom prozoru.}

**for** $i = k + lacc + 1$ **to** $\min(ul, k + lacc + nl - ul)$ **do**

$A_{1:m,i} \leftrightarrow A_{1:m,nl+k+lacc-i}$

**end for**

$res(i) \leftarrow \|A_{k+lacc+1:m,i}\|_2$ za $i$ od $k + lacc + 1$ do $nl$

$sl \leftarrow nl - nrj$; $\quad ul \leftarrow nl$

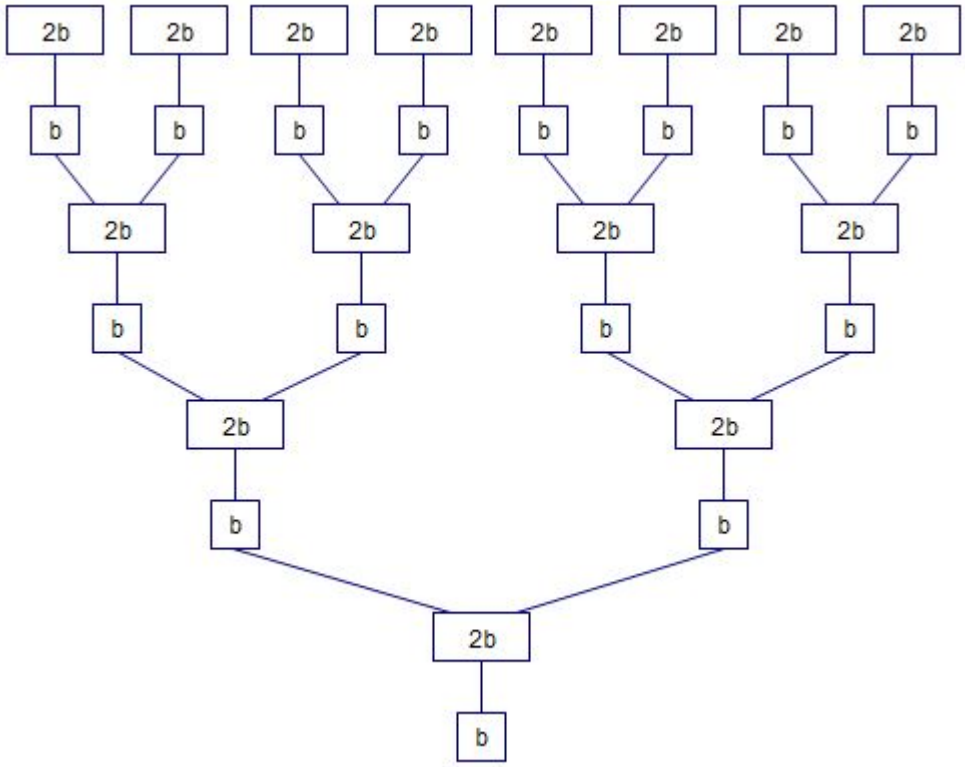
**end if**

## 2.2 Algoritam s izbjegavanjem komunikacije

Demmel, Grigori, Gu i Xiang su u [5] predložili algoritam s izbjegavanjem komunikacije. Moguće je da se algoritam koji obavlja višestruko veći ukupan broj operacija izvrši brže zbog manje komunikacije između brze i spore memorije, ili, u slučaju paralelnog računala, zbog bolje raspodjele posla na procesore.

Matricu dimenzija $m \times n$ dijelimo na $n/(2b)$ blokova dimenzija $m \times 2b$. Cilj je redukcijom dobiti jedan blok od $b$ stupaca koji će biti prvi kandidati za pivotiranje. To činimo tako da na svakom bloku izvodimo QRCP faktorizaciju kako bismo od $2b$ stupaca odabrali $b$ stupaca koji odlaze na sljedeću razinu redukcije. Po 2 bloka grupiramo zajedno i od

$b$ stupaca iz jednog i drugog bloka načinimo novi blok dimenzija $m \times 2b$ te na novim blokovima nastavljamo s algoritmom. Na kraju ostaje jedan blok od $2b$ stupaca na kojem QRCP faktorizacijom odabiremo $b$ najboljih kandidata za početne pivotne stupce. Time smo našli prvih $b$ pivotnih stupaca. Postupak ponavljamo i svakom njegovom primjenom određujemo novih $b$ pivotnih stupaca dok ne obavimo QRCP faktorizaciju do kraja ili ne dođemo do željenog broja stupaca.




Slika 2.1: Redukcija $16b$ kandidata na konačnih $b$ pivotnih stupaca

Podjela matrice na dovoljno male blokove omogućava spremanje cijele matrice u bržu memoriju, a na paralelnim računalima se redukcija na različitim granama može računati istovremeno, što donosi ubrzanje, iako je ukupan broj obavljenih računskih operacija veći nego u slučaju klasične QRCP faktorizacije. U slučaju vrlo visokih matrica teško je naći dovoljno mali $b$ tako da blokovi veličine $n \times 2b$ stanu u brzu memoriju. Za takve se blokove koriste algoritmi prilagođeni tankim i visokim matricama. Često se za matrice $A \in \mathbb{R}^{m \times n}$ gdje je $m \gg n$ prvo radi obična QR faktorizacija matrice $A$ i zatim QRCP faktorizacija matrice $R \in \mathbb{R}^{n \times n}$ znatno manje visine. Detaljniji opis takvih algoritama nalazi se u [6].

## 2.3 Randomizirana QR faktorizacija

Algoritam randomizirane QR faktorizacije prvi su predstavili Martinsson u [9], te Duersch i Gu u [7]. Randomizaciju su i prije neki autori koristili u numeričkoj linearnoj algebri, npr. u [8] se koristi randomizacija za određivanje aproksimacija matricama nižeg ranga. Matricu $A \in \mathbb{R}^{m\times n}$ množimo slijeva slučajnom matricom $\Omega \in \mathbb{R}^{l\times m}$ gdje je $l \ll m$. Tako dobivamo znatno manju matricu dimenzija $l \times n$ čiji su retci linearne kombinacije redaka matrice $A$. Stupci matrice $\Omega A$ u pravilu imaju vrlo slične linearne ovisnosti kao stupci matrice $A$. Na matrici $\Omega A$ stoga možemo primijeniti QRCP algoritam i pomoću njega dobiti redoslijed stupaca za pivotiranje koji primijenjujemo na matrici $A$ što nam omogućava primjenu algoritma za QR faktorizaciju bez pivotiranja. Slučajne elemente matrice $\Omega$ biramo nezavisno iz normalne distribucije, npr. $N(0, 1)$. Broj $l$ moramo odabrati tako da bude veći ili jednak $n$, jer inače pivotiranje nećemo moći provesti na svim stupcima. Obično se uzima $l = n + p$, gdje je $p$ broj dodatnih redaka u uzorku (engl. over-sampling parameter). Za $l$ možemo uzeti neki konstantni mali prirodni broj, npr. 5 i 10 su se pokazali kao dobri izbori. U [9] i u slučaju da tražimo aproksimaciju matrice matricom ranga $r$, možemo uzeti $l = r + p$ jer pivotiranje provodimo samo na prvih $r$ stupaca.

**Algoritam 10** Randomizirana QRCP faktorizacija s jednim uzorkovanjem

**Require:** $A \in \mathbb{R}^{m\times n}$
**Require:** $r$ je traženi aproksimacijski rang, $r \leq n$
{U slučaju da trebamo punu QRCP faktorizaciju $r = n$}
**Require:** $p$ je broj dodatnih redaka u uzorku
$l = k + p$
Generiraj slučajnu matricu $\Omega \in \mathbb{R}^{l\times m}$
$B = \Omega A$
Nađi QRCP faktorizaciju $B = QRP^T$
{U slučaju $r < n$ izvršava se skraćena QRCP faktorizacija}
Primijeni dobivenu permutaciju $A' \leftarrow AP$
Izvrši QR faktorizaciju na prvih $r$ stupaca matrice $A'$, odredi kompoziciju pripadnih $r$ reflektora.
Primijeni prethodno dobivene reflektore na ostatak matrice $A'$.
$Q$ je kopmozicija svih dobivenih reflektora.
$R = A'$

Algoritam 10 je koristan za jako uske matrice ili za jako mali aproksimacijski rang $r$. U [7] primijećeno je da se on može dobro kombinirati s algoritmom iz 2.2. Naime, algoritam 10 možemo koristiti unutar algoritma s izbjegavanjem komunikacije kako bismo faktorizirali uske blokove dimenzija $m \times 2b$.

U slučaju pune QRCP faktorizacije relativno široke matrice algoritam ne donosi veliko poboljšanje. U slučaju kvadratne matrice $A$, matrica $B$ će imati više redaka od $A$ i biti još zahtjevnija za QRCP faktorizaciju. Postavlja se pitanje kako iskoristiti randomizaciju u tim slučajevima. U literaturi se najčeše spominje blokovski algoritam s ponavljanim uzorkovanjem. Za neku odabranu širinu bloka $b$ se $n/b$ puta izvršava algoritam 10 te svakom njegovom primjenom dobivamo novih $b$ pivotnih stupaca.

**Algoritam 11** Randomizirana QRCP faktorizacija s ponavljanim uzorkovanjem

**Require:** $A \in \mathbb{R}^{m\times n}$
**Require:** $r$ je traženi aproksimacijski rang, $r \leq n$
**Require:** $p$ je broj dodatnih redaka u uzorku
**Require:** $b$ je veličina bloka
$l = k + p$
**for** $j = 1, 2, \ldots \dfrac{r}{b}$ **do**
Izvrši algoritam 10 na matrici $A_{(j-1)b+1:m,(j-1)b+1:n}$ za aproksimacijski rang $r = b$.
**end for**
$Q$ je produkt matrica $Q^{[1]}\, Q^{[2]} \ldots Q^{[k/b]}$

Duersch i Gu su u [7] predstavili drukčiju verziju blokovske randomizirane QRCP faktrizacije koja izbjegava ponovno generiranje slučajne matrice $\Omega$ u svakom koraku i množenje njome. Za početak ćemo dati nekoliko činjenica vezanih za algoritam.

**Teorem 2.3.1.** *Neka je $\Omega \in \mathbb{R}^{l\times m}$ slučajna matrica s nezavisnim elementima iz normalne razdiobe $N(0,1)$. Neka je $Q \in \mathbb{R}^{l\times l}$ ortogonalna matrica izabrana neovisno o $\Omega$. Tada je $Q\Omega \in \mathbb{R}^{l\times m}$ također slučajna matrica s nezavisnim elementima iz normalne razdiobe $N(0,1)$.*

**Teorem 2.3.2.** *Neka je $\Omega \in \mathbb{R}^{l\times m}$ slučajna matrica s nezavisnim elementima iz normalne razdiobe $N(0,1)$. Neka je $Q \in \mathbb{R}^{m\times m}$ ortogonalna matrica izabrana neovisno o $\Omega$. Tada je $\Omega Q \in \mathbb{R}^{l\times m}$ također slučajna matrica s nezavisnim elementima iz normalne razdiobe $N(0,1)$.*

Dakle, ako slučajnu matricu s nezavisnim elementima iz normalne razdiobe $N(0,1)$ pomnožimo slijeva ili zdesna kvadratnom ortogonalnom matricom koja je izabrana neovisno o $\Omega$ onda opet dobivamo slučajnu matricu s nezavisnim elementima iz normalne razdiobe $N(0,1)$. U algoritmu 10 skraćenom QRCP faktorizacijom matrice $B$ dobivamo

$$BP = Q_b \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix},$$

gdje je $S_{11} \in \mathbb{R}^{m \times n}$ gornjetrokutasta. Također, QR faktorizacijom matrice $AP$ dobivamo

$$AP = Q_a \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix},$$

gdje je $R_{11}$ gornjetrokutasta.

Sada definiramo

$$W := Q_b^T \Omega Q_a.$$

Ako matricu $W$ zapišemo kao blok-matricu,

$$W = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix},$$

dobivamo da je

$$\Omega = Q_b \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} Q_a^T.$$

Zbog $B = \Omega A$ dalje zaključujemo:

$$\begin{aligned} BP &= \Omega AP, \\ Q_b S &= Q_b W Q_a^T Q_a R, \\ S &= WR, \\ \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix} &= \begin{bmatrix} W_{11}R_{11} & W_{11}R_{12} + W_{12}R_{22} \\ W_{21}R_{11} & W_{21}R_{12} + W_{22}R_{22} \end{bmatrix}. \end{aligned}$$

Ako je $S_{11} \in \mathbb{R}^{b \times b}$ nesingularna matrica tada su i $W_{11}$ i $R_{11}$ nesingularne, te je $W_{11} = S_{11}R_{11}^{-1}$ gornjetrokutasta i $W_{21} = 0R_{11}^{-1} = 0$. $R_{22}$ je matrica na kojoj želimo nastaviti blokovski algoritam pa ćemo je odsad označavati sa $A'$. Tada možemo pisati:

$$\begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} \\ 0 & W_{22} \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ 0 & A' \end{bmatrix}.$$

Za sljedeću slučajnu matricu sada uzimamo

$$\Omega' = \begin{bmatrix} W_{12} \\ W_{22} \end{bmatrix}.$$

Po teoremu 2.3.2 može se činiti da je $\Omega'$ slučajna matrica s nezavisnim elementima iz normalne razdiobe $N(0, 1)$ jer je $W = Q_b^T \Omega Q_a$, a $Q_b$ i $Q_a$ su ortogonalne te je $\Omega'$ podmatrica matrice $W$. Međutim, $Q_a$ i $Q_b$ ne zadovoljavaju uvjet teorema jer nisu izabrane neovisno o $\Omega$. Zato teorem 2.3.2 ne garantira da je ovako definirana matrica $\Omega$ dobar izbor, ali

testiranja na računalu pokazuju da taj izbor daje dobar izbor pivotnih stupaca, usporediv s onim u algoritmu 11.

Uz tako odabranu slučajnu matricu $\Omega'$ u sljedećem koraku dobivamo:

$$B' = \Omega' A' = \begin{bmatrix} S_{12} - W_{11}R_{12} \\ S_{22} \end{bmatrix} = \begin{bmatrix} S_{12} - S_{11}R_{11}^{-1}R_{12} \\ S_{22} \end{bmatrix},$$

što omogućava određivanje matrice $B'$ bez eksplicitnog kreiranja matrice $\Omega'$. Sada opisujemo blokovski algoritam za randomiziranu QRCP koji koristi prijašnja razmatranja.

---

**Algoritam 12** Randomizirana QRCP faktorizacija s ponovnim korištenjem početnog uzorka

---

**Require:** $A \in \mathbb{R}^{m \times n}$
**Require:** $r$ je traženi aproksimacijski rang, $r \leq n$
**Require:** $p$ je broj dodatnih redaka u uzorku
**Require:** $b$ je veličina bloka
  $l = k + p$
  Generiraj slučajnu matricu $\Omega \in \mathbb{R}^{l \times m}$
  $B = \Omega A$
  **for** $j = 1, 2, \ldots \dfrac{r}{b}$ **do**
    Nađi QRCP faktorizaciju $B = QRP^T$
    Primijeni dobivenu permutaciju na $A_{1:m,(j-1)b+1:n} \leftarrow A_{1:m,(j-1)b+1:n}P$
    Izvrši QR faktorizaciju na matrici $A_{(j-1)b+1:m,(j-1)b+1:jb}$, odredi kompoziciju pripadnih $r$ reflektora.
    Primijeni prethodno dobivene reflektore na ostatak matrice: $A_{jb+1:m,jb+1:n}$.
    $B \leftarrow \begin{bmatrix} S_{12} - S_{11}R_{11}^{-1}R_{12} \\ S_{22} \end{bmatrix}$ kako je opisano u 2.3.
  **end for**
  $Q$ je kompozicija svih primijenjenih Householderovih reflektora
  $R = A$
  $P$ je kompozicija svih primijenjenih permutacija.

---

Jedan mogući problem ovog algoritma je slučaj singularne matrice $R_{11}$ što se može dogoditi ako je početna matrica $A$ singularna. Zbog toga je algoritam 11 vjerojatno nešto robusniji.

Osim navedenih algoritama u [7] se spominje izmijenjena verzija algoritma 12, pogodna za skraćenu QRCP faktorizaciju sa željenim aproksimacijskim rangom $r$ puno manjim od širine matrice $A$. Pritom se u svakom koraku izbjegava ažuriranje matrice $A$ novim blok-reflektorom. U [9] i [7] se razvijaju i daljnji algoritmi koji pokušavaju aproksimirati SVD matrice.

# Poglavlje 3

# Usporedba algoritama

Algoritmi su implementirani i testirani na Intelovom računalu Xeon Phi sa 64 jezgre i 1 MB L2 cache memorije. Testiranja će pokazati uspješnost korištenja memorijske hijerarhije i paralelizacije algoritama. Testirat ćemo vrijeme izvršavanja algoritama ovisno o dimenzijama matrice i broju korištenih procesorskih jezgri. Provjerit ćemo i uspješnost faktorizacija u aproksimacijama nižeg ranga te ćemo usporediti poredak izabranih pivotnih stupaca na raznim slučajnim matricama. Pokazat ćemo i aproksimacije nižeg ranga na primjeru crno-bijele slike.

U ovom poglavlju usporedit ćemo potprograme iz biblioteke LAPACK i to `dgeqrf` za QR faktorizaciju koja koristi algoritam 3, `dgeqp3` za QRCP faktorizaciju koja koristi algoritam 6, s implementacijom algoritama 11 i 12 dostupnima na GitHub repozitoriju (`https://github.com/ivceh/QR_faktorizacija`). Navedena 4 algoritma redom ćemo označavati s `dgeqrf`, `dgeqp3`, `rsrqrcp` i `rqrcp`.

## 3.1 Vremenska efikasnost algoritama

Za početak testiramo vremenske performanse navedenih algoritama za različite matrice i brojeve korištenih jezgri procesora. Korištene su različite slučajne matrice o kojima će kasnije biti više riječi. Pokazalo se da su za trajanje izvršavanja algoritma bitne samo dimenzije matrice. Pritom distribucija elemenata tih matrica nema značajan utjecaj. Stoga ćemo prikazati vremena izvršavanja dobivena samo korištenjem matrica s elementima iz uniformne razdiobe $U(-1, 1)$. Za parametre randomizirane QRCP faktorizacije koristili smo $b = 512$ i $p = 10$, koji su se pokazali dobrima pri testiranju.

U tablici 3.1 vidimo kako promjena broja procesorskih jezgri utječe na trajanje algoritma. Vidljivo je kako `dgeqrf` najbolje iskorištava paralelizam: prelaskom s 1 na 64 procesorske jezgre algoritam se ubrza skoro 30 puta.

Potprogram `dgeqp3` najgore iskorištava paralelizam računala: prelaskom s 1 na 64 jezgre dobiva ubrzanje od samo 5 puta. Algoritmi `rsrqrcp` i `rqrcp` su znatno bolji od `dgeqp3` po vremenu izvršavanja i po iskorištavanju paralelizma, a po performansama približavaju se potprogramu `dgeqrf`.

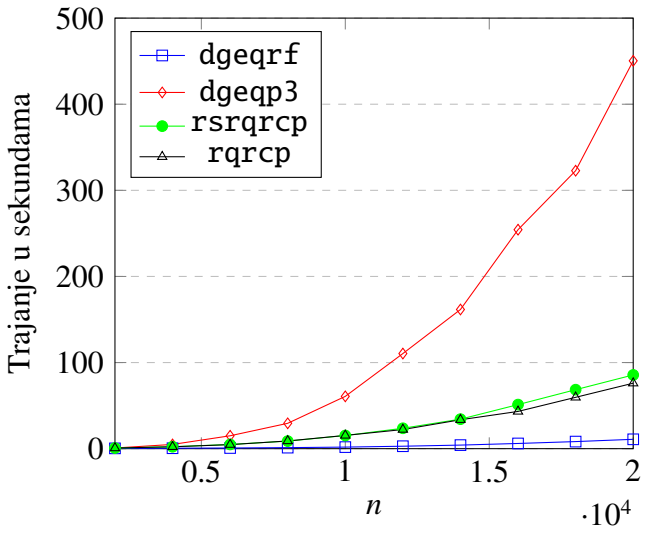
| **broj dretvi** | **dgeqrf** | **dgeqp3** | **rsrqrcp** | **rqrcp** |
|---|---|---|---|---|
| 1 | 54.624 | 281.371 | 129.183 | 98.558 |
| 2 | 32.064 | 343.977 | 86.271 | 66.424 |
| 4 | 15.206 | 176.657 | 48.643 | 38.714 |
| 8 | 8.352 | 114.069 | 30.873 | 26.882 |
| 16 | 4.452 | 98.796 | 22.873 | 20.491 |
| 32 | 2.603 | 63.556 | 16.709 | 15.356 |
| 64 | 1.832 | 55.409 | 15.591 | 14.825 |

Tablica 3.1: Vrijeme izvršavanja (u sekundama) algoritama na matrici dimenzija $10000 \times 10000$ ovisno o broju dretvi.

Na sljedećim slikama vidimo ovisnost vremena izvršavanja o dimenzijama matrice. Programi su izvršeni korištenjem sve 64 procesorske jezgre. Na slici 3.1 vidimo ponašanje algoritma za kvadratne matrice, dok je na slici 3.2 grafički prikaz vremena izvršavanja.

| **dimenzije** | **dgeqrf** | **dgeqp3** | **rsrqrcp** | **rqrcp** |
|---|---|---|---|---|
| $2000 \times 2000$ | 0.362 | 0.753 | 0.741 | 0.738 |
| $4000 \times 4000$ | 0.536 | 4.926 | 2.236 | 2.195 |
| $6000 \times 6000$ | 0.787 | 14.974 | 4.901 | 4.686 |
| $8000 \times 8000$ | 1.202 | 29.445 | 8.778 | 8.941 |
| $10000 \times 10000$ | 1.827 | 60.741 | 15.271 | 15.380 |
| $12000 \times 12000$ | 2.847 | 110.450 | 23.683 | 22.216 |
| $14000 \times 14000$ | 4.153 | 161.614 | 34.257 | 33.571 |
| $16000 \times 16000$ | 6.046 | 254.305 | 51.164 | 43.249 |
| $18000 \times 18000$ | 8.221 | 322.791 | 68.489 | 59.626 |
| $20000 \times 20000$ | 10.873 | 450.383 | 85.623 | 76.029 |

Slika 3.1: Brzina izvršavanja algoritama (u sekundama) na matricama dimenzija $n \times n$.



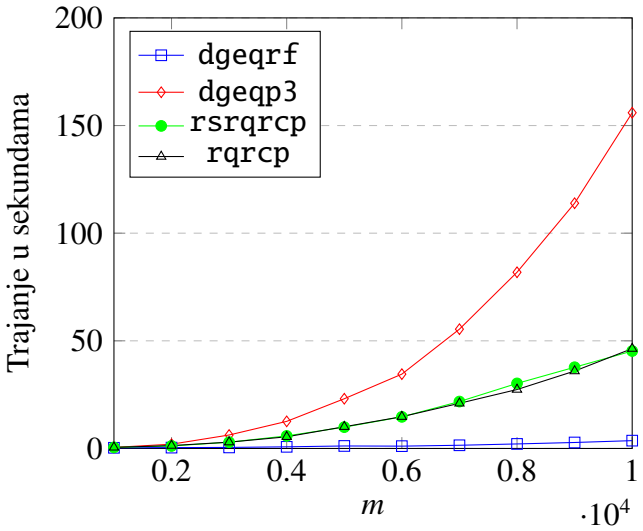

Slika 3.2: Grafički prikaz brzine izvršavanja algoritama na matricama dimenzija $n \times n$.

Na slici 3.3 (grafički prikaz je na slici 3.4) su algoritmi testirani na matricama dimenzija $m \times 2m$.

| **dimenzije** | **dgeqrf** | **dgeqp3** | **rsrqrcp** | **rqrcp** |
|---|---|---|---|---|
| $1000 \times 2000$ | 0.307 | 0.498 | 0.496 | 0.507 |
| $2000 \times 4000$ | 0.397 | 1.984 | 1.162 | 1.341 |
| $3000 \times 6000$ | 0.488 | 6.253 | 2.910 | 2.930 |
| $4000 \times 8000$ | 0.765 | 12.589 | 5.829 | 5.420 |
| $5000 \times 10000$ | 1.163 | 23.095 | 9.906 | 10.089 |
| $6000 \times 12000$ | 1.082 | 34.531 | 14.682 | 14.759 |
| $7000 \times 14000$ | 1.486 | 55.419 | 21.789 | 20.981 |
| $8000 \times 16000$ | 2.106 | 81.844 | 30.213 | 27.378 |
| $9000 \times 18000$ | 2.778 | 113.919 | 37.731 | 35.957 |
| $10000 \times 20000$ | 3.636 | 155.941 | 45.253 | 46.385 |

Slika 3.3: Brzina izvršavanja algoritama (u sekundama) na matricama dimenzija $m \times 2m$.

Na slici 3.5 (grafički prikaz je na na slici 3.6) su algoritmi testirani na visokim i tankim matricama dimenzija $2n \times n$. Testiranja pokazuju da je `dgeqrf` i dalje mnogo efikasnija od

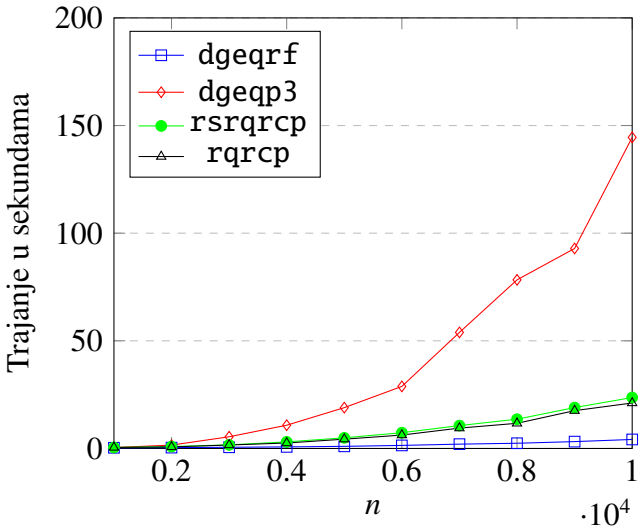



Slika 3.4: Grafički prikaz brzine izvršavanja algoritama na matricama dimenzija $m \times 2m$.

QRCP faktorizacije. Randomizirana QRCP faktorizacija za dovoljno velike matrice uvijek ima višestruko bolje performanse od klasične QRCP faktorizacije.

| **dimenzije** | **dgeqrf** | **dgeqp3** | **rsrqrcp** | **rqrcp** |
|---|---|---|---|---|
| 2000 × 1000 | 0.321 | 0.420 | 0.465 | 0.470 |
| 4000 × 2000 | 0.401 | 1.564 | 0.889 | 0.719 |
| 6000 × 3000 | 0.555 | 5.432 | 1.659 | 1.610 |
| 8000 × 4000 | 0.731 | 10.806 | 3.056 | 2.532 |
| 10000 × 5000 | 0.985 | 18.959 | 4.942 | 4.319 |
| 12000 × 6000 | 1.421 | 28.826 | 7.323 | 6.222 |
| 14000 × 7000 | 2.009 | 53.906 | 10.626 | 9.464 |
| 16000 × 8000 | 2.424 | 78.316 | 13.598 | 11.683 |
| 18000 × 9000 | 3.186 | 92.899 | 18.998 | 17.570 |
| 20000 × 10000 | 4.185 | 144.500 | 23.662 | 21.122 |

Slika 3.5: Brzina izvršavanja algoritama (u sekundama) na matricama dimenzija $2n \times n$.




Slika 3.6: Grafički prikaz brzine izvršavanja algoritama na matricama dimenzija $2n \times n$.

## 3.2 Aproksimacije nižeg ranga

U ovom odjeljku testirat ćemo koliko se dobro implementirani algoritmi mogu primijeniti za aproksimaciju matrice matricama nižeg ranga. Proučavat ćemo Frobeniusovu normu razlike matrice $A$ i njezine aproksimacije matricom zadanog ranga $r$. Korištenjem dekompozicije singularnih vrijedosti (SVD) dobit ćemo matricu ranga $r$ koja najbolje aproksimira zadanu matricu. Točnosti aproksimacija dobivenih ostalim aproksimacijama: skraćenom QR faktorizacijom, kao što je to opisano u poglavlju 1.3, klasičnom QRCP faktorizacijom i randomiziranim QRCP faktorizacijama usporedit ćemo s točnošću najbolje aproksimacije dobivene SVD-om.

U ovom testiranju je bitno kako generiramo matricu $A$, jer to određuje blizinu najbliže aproksimacije nižeg ranga. Prvi primjeri bit će slučajne matrice s nezavisnim elementima iz uniformne razdiobe $U(-1,1)$ i normalne razdiobe $N(0,1)$.

U tablicama 3.2 i 3.3 vidimo da svim vrstama QR faktorizacije dobivamo poprilično lošu aproksimaciju nižeg ranga. Stupčano pivotiranje tijekom QR faktorizacije ne donosi znatno bolju aproksimaciju od uzimanja prvih $r$ stupaca iz obične QR faktorizacije.

Očekujemo bolji uspjeh na nekim drugim, ”pravilnijim” matricama.

| **rang** | **200** | **400** | **600** | **800** |
|---|---|---|---|---|
| **dgeqrf** | 800.066 | 599.625 | 399.991 | 199.233 |
| **dgeqp3** | 799.999 | 593.305 | 388.563 | 187.593 |
| **rsrqrcp** | 796.383 | 594.685 | 394.737 | 197.582 |
| **rqrcp** | 796.194 | 593.927 | 394.684 | 196.775 |
| **SVD** | 680.993 | 431.631 | 230.732 | 80.714 |

| **rang** | **900** | **950** | **990** |
|---|---|---|---|
| **dgeqrf** | 99.134 | 49.237 | 10.035 |
| **dgeqp3** | 89.386 | 41.896 | 6.402 |
| **rsrqrcp** | 98.581 | 49.553 | 8.443 |
| **rqrcp** | 98.523 | 48.709 | 9.131 |
| **SVD** | 28.123 | 9.740 | 0.863 |

Tablica 3.2: Greške u aproksimaciji matricom nižeg ranga matrice $A \in \mathbb{R}^{1000\times1000}$ s nezavisnim elementima iz normalne distribucije $N(0,1)$, $\|A\|_F = 1000.36$.

| **rang** | **200** | **400** | **600** | **800** |
|---|---|---|---|---|
| **dgeqrf** | 461.868 | 346.585 | 231.207 | 115.809 |
| **dgeqp3** | 461.868 | 343.267 | 225.596 | 109.676 |
| **rsrqrcp** | 460.667 | 344.675 | 229.221 | 114.827 |
| **rqrcp** | 460.356 | 344.138 | 228.751 | 114.387 |
| **SVD** | 393.234 | 249.297 | 133.575 | 46.958 |

| **rang** | **900** | **950** | **990** |
|---|---|---|---|
| **dgeqrf** | 58.107 | 29.290 | 5.536 |
| **dgeqp3** | 52.909 | 25.164 | 3.478 |
| **rsrqrcp** | 57.566 | 28.800 | 5.943 |
| **rqrcp** | 57.188 | 28.784 | 6.057 |
| **SVD** | 16.697 | 5.911 | 0.470 |

Tablica 3.3: Greške u aproksimaciji matricom nižeg ranga matrice $A \in \mathbb{R}^{1000\times1000}$ s nezavisnim elementima iz normalne distribucije $U(-1,1)$, $\|A\|_F = 577.036$.

Slijedeći je test na primjerima s predodređenim singularnim vrijednostima s eventualnom malom slučajnom perturbacijom elemenata. Takve matrice konstruirane su tako da se dijagonalna matrica, sa željenim singularnim vrijednostima na dijagonali, pomnoži slijeva i/ili zdesna sa slučajnim ortogonalnim matricama. Parametar $p$ ni ovdje nije pokazao značajni utjecaj, pa smo za testiranje uzeli vrijednost $p = 10$.

| **rang** | **250** | **290** | **300** | **310** | **350** |
|---|---|---|---|---|---|
| **dgeqrf** | 710.096 | 345.170 | 224.218 | 143.146 | 68.891 |
| **dgeqp3** | 709.038 | 325.615 | 103.099 | 75.591 | 53.152 |
| **rsrqrcp** | 709.071 | 326.266 | 105.667 | 78.057 | 53.640 |
| **rqrcp** | 709.105 | 326.363 | 108.452 | 78.842 | 53.932 |
| **SVD** | 707.602 | 317.333 | 26.457 | 26.268 | 25.495 |

Tablica 3.4: Greške u aproksimaciji matricom nižeg ranga slučajne matrice $A \in \mathbb{R}^{1000\times1000}$ s predodređenim slučajnim vrijednostima 100 (s kratnošću 300) i 1 (s kratnošću 700), $||A||_F = 1732.25$.

| **rang** | **250** | **290** | **300** | **310** | **350** |
|---|---|---|---|---|---|
| **dgeqrf** | 1776.56 | 815.965 | 518.773 | 340.041 | 168.566 |
| **dgeqp3** | 1773.13 | 841.433 | 552.684 | 387.115 | 171.540 |
| **rsrqrcp** | 1687.56 | 720.051 | 268.618 | 205.007 | 139.574 |
| **rqrcp** | 1692.49 | 730.851 | 278.084 | 208.253 | 139.619 |
| **SVD** | 1031.53 | 375.268 | 70.056 | 68.146 | 61.239 |

Tablica 3.5: Greške u aproksimaciji matricom nižeg ranga slučajne matrice $A \in \mathbb{R}^{1000\times1000}$ dobivene kao $BC + 0.1N$ gdje su $B \in \mathbb{R}^{1000\times300}$ i $C \in \mathbb{R}^{300\times1000}$ slučajne matrice s nezavisnim elementima iz razdiobe $U(-1, 1)$ i $C \in \mathbb{R}^{1000\times1000}$ slučajna matrica s nezavisnim elementima iz razdiobe $N(0, 1)$, $||A||_F = 5768.52$.

U tablicama 3.4, 3.5 i 3.6 vidimo kako je QRCP faktorizacija uglavnom dala značajno bolje aproksimacije od QR faktorizacije na različitim matricama dimenzija $1000 \times 1000$ koje se mogu dobro aproksimirati matricom ranga 300. Randomizirana QRCP faktorizacija je u sva 3 testa dala rezultate usporedive s klasičnom QRCP faktorizacijom.

| rang | 250 | 290 | 300 | 310 | 350 |
|---|---|---|---|---|---|
| `dgeqrf` | 5550.19 | 4638.40 | 4754.53 | 4718.09 | 2073.17 |
| `dgeqp3` | 1314.44 | 761.37 | 757.73 | 797.87 | 210.31 |
| `rsrqrcp` | 615.25 | 278.76 | 293.41 | 305.69 | 74.94 |
| `rqrcp` | 399.62 | 219.20 | 227.51 | 226.00 | 72.92 |
| **SVD** | 186.93 | 148.37 | 148.36 | 148.39 | 65.59 |

Tablica 3.6: Greške u aproksimaciji matricom nižeg ranga matrice $A \in \mathbb{R}^{1000\times1000}$ dobivene kao $DQ + N$ gdje su $D, Q, N \in \mathbb{R}^{1000\times1000}$, $D$ dijagonalna s nulama na prvih 700 mjesta na dijagonali i uzastopne višekratnike od 10 na sljedećih 300 mjesta $(10, 20, 30, \ldots, 3000)$, $Q$ slučajna ortogonalna i $N$ slučajna matrica s nezavisnim elementima iz normalne distribucije $N(0, 1)$, $\|A\|_F = 30075.1$.

## Aproksimacije slike

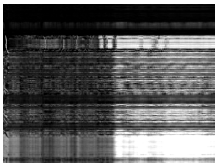
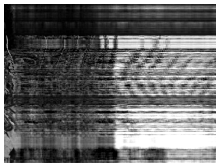
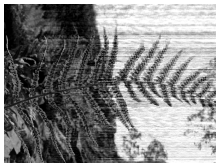
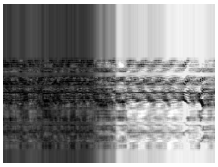
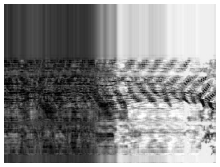
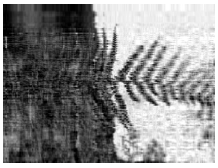
Vrlo dobar primjer matrice koja se može dobro aproksimirati matricom nižeg ranga je matrica piksela slike ili fotografije. Naime, u pravilu slike imaju mnogo sličnih stupaca, a posebno se sličnost može uočiti na susjednim i bliskim stupcima piksela. Aproksimacija takve matrice matricom nižeg ranga može biti korisna u nekim slučajevima. Jedna moguća primjena je sažimanje slike u manju memoriju. Npr. ako smo za aproksimaciju slike dimenzija $1000\times1000$ matricom ranga 100 koristili QRCP faktorizaciju, tada je za rekonstrukciju aproksimacije dovoljno zapamtiti $Q_{1:1000,1:100}$, $R_{1:100,1:1000}$ i vektor permutacije $P$. Time smo gotovo 5 puta smanjili količinu brojeva koje moramo pamtiti kako bismo konstruirali sliku, što može donijeti memorijske uštede. Ako je aproksimacija dovoljno bliska, promatrač ne bi trebao vidjeti tu malu razliku u nijansi pojedinih piksela. Aproksimacija nižeg ranga može biti korisna i u području strojnog učenja, jer na njoj sitni detalji često gube izražaj, što pomaže algoritmima strojnog učenja u prepoznavanju bitnijih karakteristika slike.

U našem slučaju korištena je crno–bijela fotografija paprati dimenzija $1024 \times 768$, koju smo pokušali aproksimirati matricama ranga 25, 50, 100 i 200. S obzirom da se radi o crno–bijeloj slici, za njezin prikaz u računalu dovoljna je jedna matrica prirodnih brojeva od 0 do 255 (inače bi trebale 3 takve matrice za prikaz boja u RGB modelu). U tablici 3.7 vidimo točnost aproksimacija matricama nižeg ranga, a u tablici 3.8 vidimo kako te aproksimacije izgledaju. Opet se vidi veća točnost QRCP algoritama od obične QR faktorizacije i manja točnost od SVD-a. Randomizirani QRCP algoritmi su, na ovom primjeru, dali i bolje rezultate od klasične QRCP faktorizacije.

Slika 3.7: Crno-bijela slika korištena kao testni primjer, preuzeto sa [4].

| **rang** | **25** | **50** | **100** | **200** |
|---|---|---|---|---|
| **dgeqrf** | 89840.9 | 76875.0 | 51244.5 | 26215.50 |
| **dgeqp3** | 45387.7 | 40997.2 | 18905.9 | 8972.81 |
| **rsrqrcp** | 29144.4 | 22596.7 | 15097.0 | 8236.49 |
| **rqrcp** | 29070.6 | 22193.4 | 15399.1 | 8146.95 |
| **SVD** | 22409.5 | 16269.2 | 10064.0 | 5073.82 |

Tablica 3.7: Greške u aproksimaciji matricom nižeg ranga matrice $A \in \mathbb{R}^{768\times1024}$ popunjena pikselima slike 3.7, $\|A\|_F = 138472$.

| **rang** | **25** | **50** | **100** | **200** |
|---|---|---|---|---|
| **dgeqrf** |  |  |  |  |
| **dgeqp3** |  |  |  |  |
| **rsrqrcp** |  |  |  |  |
| **rqrcp** |  |  |  |  |
| **SVD** |  |  |  |  |

Tablica 3.8: Slike konstruirane iz slike 3.7 metodama aproksimacije matricama nižeg ranga.

# Bibliografija


[1] Christian H. Bischof, *Incremental Condition Estimation*, SIAM Journal on Matrix Analysis and Applications **11** (1990), br. 2, 312–322, ISSN 0895-4798, http://epubs.siam.org/doi/10.1137/0611021.

[2] Christian Bischof, *A Parallel QR Factorization Algorithm with Controlled Local Pivoting*, SIAM Journal on Scientific and Statistical Computing **12** (1991), br. 1, 36–57, ISSN 0196-5204, http://epubs.siam.org/doi/10.1137/0912002.

[3] Christian Bischof i Charles Van Loan, *The WY Representation for Products of Householder Matrices*, SIAM Journal on Scientific and Statistical Computing **8** (1987), br. 1, s2–s13, ISSN 0196-5204, http://epubs.siam.org/doi/10.1137/0908009.

[4] Marcus Buckwald, *Slika, licenca https://creativecommons.org/licenses/by-sa/2.0/*, 2018, https://www.flickr.com/photos/marcus-buchwald/35961374501/sizes/l, posjećena 2018-11-10.

[5] James W. Demmel, Laura Grigori, Ming Gu i Hua Xiang, *Communication Avoiding Rank Revealing QR Factorization with Column Pivoting*, SIAM Journal on Matrix Analysis and Applications **36** (2015), br. 1, 55–89, ISSN 0895-4798, http://epubs.siam.org/doi/10.1137/13092157X.

[6] James Demmel, Laura Grigori, Mark Hoemmen i Julien Langou, *Communication-optimal Parallel and Sequential QR and LU Factorizations*, SIAM Journal on Scientific Computing **34** (2012), br. 1, A206–A239, ISSN 1064-8275, http://epubs.siam.org/doi/10.1137/080731992.

[7] Jed A. Duersch i Ming Gu, *Randomized QR with Column Pivoting*, SIAM Journal on Scientific Computing **39** (2017), br. 4, C263–C291, ISSN 1064-8275, http://epubs.siam.org/doi/10.1137/15M1044680.

[8] N. Halko, P. G. Martinsson i J. A. Tropp, *Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions*, SIAM

Review **53** (2011), br. 2, 217–288, ISSN 0036-1445, http://epubs.siam.org/doi/10.1137/090771806.

[9] P G Martinsson, *Blocked rank-revealing QR factorizations: How randomized sampling can be used to avoid single-vector pivoting*, 2015, http://arxiv.org/abs/1505.08115, posjećeno 1. studenog 2018.

[10] Gregorio Quintana-Ortí, Xiaobai Sun i Christian H. Bischof, *A BLAS-3 Version of the QR Factorization with Column Pivoting*, SIAM Journal on Scientific Computing **19** (1998), br. 5, 1486–1494, ISSN 1064-8275, http://epubs.siam.org/doi/10.1137/S1064827595296732.

# Sažetak

U ovom radu predstavljeni su neki algoritmi za ubrzanje QR faktorizacije matrice sa stupčanim pivotiranjem (QRCP faktorizacije). QR faktorizacija se izvršava vrlo brzo na modernim računalima s memorijskom hijerarhijom i većim brojem procesoskih jezgara korištenjem blokovskog algoritma. Koristi se WY reprezentacija produkta Householderovih reflektora. Taj algoritam je implementiran u LAPACK-ovoj rutini `dgeqrf`. QRCP faktorizacija je po broju potrebnih aritmetičkih operacija neznatno zahtjevnija od QR faktorizacije, no kod nje je teško postići takvo ubrzanje blokovskim algoritmom, jer ne možemo unaprijed donijeti odluku o sljedećih $b$ pivotnih stupaca. Djelomično rješenje je blokovska QRCP faktorizacija koja je implementirana u LAPACK-ovoj rutini `dgeqp3`, no ona ne daje višestruko ubrzanje zbog potrebe za ažuriranjem jednog retka u svakom koraku. Naveli smo neke primjene QRCP faktorizacije i opisali primjenu za nalaženje aproksimacija matrice nižeg ranga gdje QRCP faktorizacija često daje rezultate usporedive s optimalnom, ali računski zahtjevnijom, singularnom dekompozicijom.

Predstavili smo tri druga pristupa QRCP faktorizaciji, koja su vremenski znatno efikasnija od klasične QRCP faktorizacije, te neke njihove varijacije. Veću brzinu im najčešće daje činjenica da ne zahtijevaju uvijek odabir stupca s najvećom normom. Prvi je algoritam s kontroliranim lokalnim pivotiranjem kod kojeg je unaprijed određena najgora dopustiva uvjetovanost matrice. Tražimo najbolje pivotne kandidate u trenutnom bloku, a u slučaju da svi preostali kandidati daju lošu uvjetovanost, blok proširujemo novim stupcima. Drugi je algoritam s izbjegavanjem komunikacije, koji dijeli matricu na blokove stupaca. Na blokovima paralelno provodi QRCP faktoriazciju i redukcijom dolazi do najboljih $b$ kandidata za sljedeće pivotne stupce. Treći je algoritam randomizirane QRCP faktorizacije koja koristi množenje slijeva slučajnom matricom $\Omega$ male visine, kako bi se smanjio broj redaka. Na takvoj se matrici provodi klasična QRCP faktorizacija te se njome izabire redoslijed pivotnih stupaca prvotne matrice $A$. Ako matrica $A$ nije dovoljno uska, generiranje slučajnog uzorka treba ponavljati, što se može učiniti traženjem novih slučajnih brojeva ili korištenjem postojećeg slučajnog uzorka.

Implementirane su dvije verzije randomizirane QRCP faktorizacije, a njihovim testiranjem potvrđeno je očekivanje da daju rezultate kvalitetom usporedive s klasičnom QRCP faktorizacijom, uz višestruko vremensko ubrzanje.

# Summary

In this paper we presented some algorithms for fast QR matrix factorization with column pivoting (QRCP factorization). The QR factorization runs very fast on modern computers due to memory hierarchy and parallel execution on multiple CPU cores by using blocked algorithm. WY representation of the product of Householder reflectors is used for this purpose. The algorithm is implemented in LAPACK routine `dgeqrf`. The QRCP factorization is not much more complex than the QR factorization, compared by the number of arithmetic operations needed, but it is hard to preform into a blocked algorithm because we cannot immediately decide about the next $b$ pivot columns. Partial solution is the blocked QRCP factorization which is implemented in LAPACK routine `dgeqp3`. It does not provide multiple speed-up because it updates only one row in each step. We showed some applications of the QRCP factorization, and described the application in finding lower rank approximations where the QRCP factorization often gives results comparable to optimal, but computationally more complex, singular value decomposition.

We presented three other approaches to the QRCP factorization, that are much more time efficient than the classical QRCP factorization, and some variations of them. What makes them fast is the fact they do not require the column with the biggest norm in each step. The first algorithm is the QR factorization with controlled local pivoting, each uses predetermined worst allowed matrix condition number. We search for the best pivot candidates in a current block, until each of them causes low condition number. Then the current block is expanded by the new columns. The second one is communication avoiding algorithm, which divides matrix to column blocks on which the QRCP factorization will be done in parallel, and it finds the best choice for $b$ pivot columns. The third algorithm is the randomized QRCP factorization which multiplies matrix $A$ from the left by $\Omega$, to decrease the number of rows. On that matrix we perform the classical QRCP factorization, which is used to detemine order of pivot columns in the initial matrix $A$. If $A$ is not thin enough, randomized sampling must be repeated which can be done by repeated random number generation, or by using, already existing, random sample.

Two versions of the randomized QRCP factorization are implemented. We tested them and confirmed the expectation that their results will be comparable to the classical QRCP factorization by quality with significant speedup.

# Životopis

Ivan Čeh rođen je 21. svibnja 1994. godine u Rijeci. Svoje djetinstvo uglavnom je proveo u selu Oslići, koje se nalazi između Buzeta i Pazina. 2009. godine završio je Osnovnu školu "Vazmoslav Gržalja" u Buzetu, gdje je 2013. godine završio i Srednju školu Buzet, smjer opća gimnazija.

Tijekom osnovnoškolskog i srednjoškolskog obazovanja sudjelovao je na brojnim natjecanjima, između ostalog i matematičkim. Najznačajnija dostignuće postigao je na državnim natjecanjima iz matematike, gdje je dva puta osvojio 3. nagradu, jednom 2. nagradu u B kategoriji i tri puta 1. nagradu u B kategoriji. Sudjelovao je na jednom državnom natjecanju iz logike i dva državna natjecanja iz fizike, na kojima je osvojio jednu 3. nagradu.

Za nastavak svog školovanja odabrao je Preddiplomski sveučilišni studij matematike na Matematičkom odsjeku Prirodoslovno–matematičkog fakulteta u Zagrebu, koji je upisao 2013. godine. Diplomski sveučilišni studij Računarstvo i matematika upisao je 2016. godine.

Tijekom fakultetskog obrazovanja sudjelovao je na mnogim samostalnim i timskim natjecanjima kao što su: izborna natjecanja za Vojtěch Jarník, izborna natjecanja za IMC, Natjecanje timova studenata informatičara hrvatskih sveučilišta, CERC 2017. i Mozgalo 2018 (osvojena 2. nagrada na zadatku "Detecting packed executable files"). Osim aktivnosti na natjecanjima, tijekom petogodišnjeg školovanja na PMF-u bio je demonstrator iz kolegija "Programiranje 1", "Programiranje 2", "Linearna algebra 1", "Linearna algebra 2" i "Diskretna matematika". Sudjelovao je u volonterskim instrukcijama, radio u firmi Ericsson Nikola Tesla na timskim i samostalnim projektima, te sudjelovao kao predavač u Ljetnoj školi matematike u Roču. U sklopu studija radio je na programskim projektima koji su se, među ostalim, bavili rješavanjem sudokua i Rubikove kocke, te implementacijom računalne igre Pacman.